Submitted to
Int. J. Control

Nonlinear system identification using neural networks: from linear to nonlinear neural state space models for MIMO systems, applicable to robust controller design*

Johan Suykens, Bart De Moor† Joos Vandewalle

K.U. Leuven - Department Electrical Engineering - ESAT
Kardinaal Mercierlaan 94, B-3001 Leuven, BELGIUM

tel: 32/16/22 09 31 - fax: 32/16/22 18 55
e-mail: suykens@esat.kuleuven.ac.be

ESAT-SISTA report 40 - 1992

December 7, 1992

Abstract

We propose an algorithm for obtaining a nonlinear state space model from input-output measurements on a MIMO system. The models are parametrized by feedforward neural networks. The learning principle is to apply first e.g. a subspace algorithm in order to obtain a linear state space model and in a second step to optimize the weights of the neural network representation with the linear model as a starting point in parameter space. The cost function to be optimized here is based on the simulation result and can be measured in an arbitrarily chosen norm. From this nonlinear model a degree of distortion and parametric uncertainties with respect to a linear model can be derived. The nonlinear models are important also if one wants to apply nonlinear control strategies in discrete time.

Keywords: nonlinear state space models, multilayer feedforward neural networks, nonlinear optimization, parametric uncertainties, MIMO processes.

*This research work was carried out at the ESAT laboratory of the Katholieke Universiteit Leuven, in the framework of a Concerted Action Project of the Flemish Community, entitled *Applicable Neural Networks*. The scientific responsibility is assumed by its authors.

†Research Associate of the Belgian National Fund for Scientific Research

1 Introduction

Multilayer feedforward neural networks with one or more hidden layers possess the property that they can approximate any continuous nonlinear function arbitrarily well (Cybenko 1989, Funahashi 1989, Hornik 1989). In this paper we make use of this property in order to *parametrize* nonlinear state space models in discrete time for multivariable systems by neural networks. The development of these nonlinear techniques is motivated by the following facts

- In robust control theory one often makes assumptions about the uncertainties (structured, unstructured, parametric etc.) on the plant model (see e.g. Francis 1987, Morari 1989). However up till now no practical algorithms exists that can extract this information from identification of input-output data sets of a *real* system. There seems to exist a gap between identification and control algorithms with respect to the description of uncertainties. The identification method that we propose in this paper, obtains a nonlinear state space model that can be interpreted as a linear system with parametric uncertainties and bounds on the matrix elements of the state space description can be given.
- In nonlinear control theory one makes extensively use of nonlinear state space models (see e.g. Isidori 1985). Most results on feedback linearization, variable structure control etc. are however based on continuous time models that are obtained in an analytical way. Similar techniques in discrete time are less developed also because there is a lack of methods for obtaining nonlinear state space models in discrete time.
- Another motivation is to have better (nonlinear) predictions by means of the nonlinear models.

With respect to learning of the interconnection weights of the neural network one can either apply global optimization techniques (like e.g. genetic algorithms) or local optimization routines. The disadvantage of global optimization is that it is a slow procedure if one wants to use it for large scale optimization problems, while in the case of local optimization one needs a good starting point. This is precisely the reason why we choose here for local optimization because linear identification results based on subspace algorithms (De Moor 1991, Moonen 1989, Van Overschee 1992) can already give good models. These algorithms have advantages with respect to structure determination, numerical robustness and computational aspects. The neural network representation initially coincides then with the linear state space method and evolves to a nonlinear state space model during optimization. The cost function is based on the simulation result of interest and can be measured in any norm. Based on this model it is possible to define so called distortion values that indicate how strong the model is distorted from a nominal linear model. The nonlinear system can also be interpreted as an uncertain linear system for which we can give bounds on the coefficients. Hence some important features of our identification algorithm are

- State space modelling approach.
- Applicability to MIMO systems.
- Identifiability in any norm (e.g. l_p for $1 \leq p \leq \infty$).
- Learning rule for interconnection weights: structured estimates through simulation error approach and starting points from linear identification theory.

- Required complexity of the neural model is partially based on linear subspace algorithms
- "Hardness" of the nonlinearity of the system can be quantified by distortion value concepts.
- The neural models can be interpreted as linear systems with parametric uncertainties. A convex hull of uncertainty emerges in a natural way and can be used for robust controller design.
- Superior nonlinear prediction.

If we compare this algorithm with standard algorithms for the identification of neural networks based on input-output representations (see e.g. Chen 1990), we can state that the latter concentrate more on accelerating the learning rules with respect to the classical backpropagation algorithm, but need a trial-and-error procedure with respect to the required complexity of the network and initial weights and they are mainly suited for modelling SISO systems.

This paper is organized as follows: in Section 2 the parametrization of nonlinear state space models by multilayer neural networks is described and distortion concepts are introduced. In Section 3 an off-line learning algorithm for the interconnection weights is proposed based on the simulation error approach. Examples are given in Section 4 for a simulated system with hysteresis and two real life systems: a ball and beam system and a glass furnace.

2 Neural state space models for MIMO processes

In this section we develop nonlinear state space models parametrized by feedforward neural networks. First we consider the autonomous case in order to explain new concepts like distortion values and then we discuss full and partial parametrizations for the non-autonomous case.

2.1 Autonomous case

We are interested in a neural network representation for f in the nonlinear model

$$x_{k+1} = f(x_k) \tag{1}$$

with state $x_k \in \mathbb{R}^n$ that can be written as

$$x_{k+1} = A(x_k).x_k \tag{2}$$

We parametrize the model (1) by a feedforward neural network with one hidden layer (see Fig.1)

$$x_{k+1} = W.tanh(V.x_k) \tag{3}$$

where we restrict the synaptic weight matrices W and V to $\mathbb{R}^{n\times n}$ and $tanh$ is the hyperbolic tangent function ($tanh(x) = \frac{1-exp(-2x)}{1+exp(-2x)}$) that is taken elementwise. Remark that any continuous nonlinear function can be represented by a feedforward neural network with one or more hidden layers. However nothing is said about the number of neurons that is needed to represent such a mapping. In (3) we restrict the number of hidden neurons to n (see also Suykens 1992 for more information on the estimation of the number of hidden neurons).

Now we consider the problem where we already have a linear model of the form

$$x_{k+1} = A.x_k \tag{4}$$

and we want to learn the model (3) from (4).
We can make the following observation: suppose we take n hidden neurons and we start the weights to learn as

$$\begin{aligned} W &:= \tfrac{1}{\alpha}.A \\ V &:= \alpha.I_n \end{aligned} \tag{5}$$

then

$$x_{k+1} = \frac{1}{\alpha}A.tanh(\alpha.I_n\, x_k) \overset{\alpha\to 0}{=} A.x_k \tag{6}$$

which follows directly from a Taylor expansion of the activation function $tanh$.
The same linearization principle can be applied to the case of more than n hidden neurons and more than one hidden layer. In the case of n_h hidden neurons ($n_h > n$) and one hidden layer

$$\begin{aligned} W &:= \tfrac{1}{\alpha}.P \\ V &:= \alpha.Q \end{aligned} \tag{7}$$

where $P(n \times n_h)$ and $Q(n_h \times n)$ are chosen such that $P.Q = A$ we get

$$x_{k+1} = \frac{1}{\alpha}P.tanh(\alpha.Q\, x_k) \overset{\alpha\to 0}{=} A.x_k \tag{8}$$

In the case of a two hidden layer system with the model

$$x_{k+1} = W.tanh(V_1.tanh(V_2.x_k)) \tag{9}$$

and n neurons in each hidden layer

$$\begin{aligned} W &:= \tfrac{1}{\alpha}.R \\ V_1 &:= I_n \\ V_2 &:= \alpha.S \end{aligned} \tag{10}$$

where $R(n \times n)$ and $S(n \times n)$ are chosen such that $R.S = A$ we get

$$x_{k+1} = \frac{1}{\alpha}R.tanh(I_n.tanh(\alpha.S.x_k)) \overset{\alpha\to 0}{=} A.x_k \tag{11}$$

From now on we will only consider the one hidden layer case with n hidden neurons.

In order to write the model (3) in the form (2) we take an elementwise description of (3) and use the Einstein summation convention (see e.g. Joshi 1977), because formulas turn out to be much simpler then, especially in the non-autonomous case. Eqn. (3) can then be written as[1]

$$x^i := w^i_j.tanh(v^j_l\, x^l) \tag{12}$$

[1] The time index k is omitted here because the Einstein summation convention must not be applied to this index. For that reason the assignment ":=" is introduced in the expression.

where the indices and all other subsequent indices in this subsection run from 1 to n. With some manipulations we can write

$$\begin{aligned} x^i &:= w_j^i.tanh(v_l^j\, x^l) \\ &:= w_j^i.\frac{tanh(v_{l_1}^j\, x^{l_1})}{v_{l_2}^j\, x^{l_2}}.v_{l_3}^j\, x^{l_3} \\ &:= w_j^i\gamma_j^j v_l^j.x^l \end{aligned} \tag{13}$$

where by definition $\gamma_j^j = \frac{tanh(v_{l_1}^j\, x^{l_1})}{v_{l_2}^j\, x^{l_2}}$. If $v_l^j\, x^l = 0$ we apply de l' Hospital's rule resulting into $\gamma_j^j = 1$. Remark that the elements γ_j^j are upper and lower bounded as $0 < \gamma_j^j \leq 1$. In Fig.2 the function $k = \frac{tanh(\alpha x)}{\alpha x}$ is plotted for different values of α.
If we turn back to matrix-vector notation (13) can be written as

$$x_{k+1} = W\Gamma(x_k)V.x_k \tag{14}$$

with $A(x_k) = W\Gamma(x_k)V.x_k$. The matrices W, Γ and V contain respectively the elements w_j^i, γ_j^j and v_l^j and Γ is a diagonal matrix. The matrix $A(x_k)$ can be decomposed as

$$A(x_k) = \sum_{i=1}^{n} \gamma_i(x_k).A_i \quad A_i = w_i.((v^t)_i)^t \tag{15}$$

where w_i denotes the i-th column of W and $(v^t)_i$ the i-th column of V^t. Remark that the system is perfectly linear if $\gamma_j = 1$ $(\forall j)$. We define *distortion values* $d_j(x_k) = -log(\gamma_j(x_k))$ (*log* is taken here in order to let high values of d_j correspond to a high distortion) such that $d_j(x_k) \geq 0$ and the larger the $d_j(x_k)$ the more nonlinear distortion is into the system. The state space behaviour can best be understood in a quasi-linear approach where the eigenvalue configuration changes depending upon the state of the system (Suykens 1991).
Another important remark is that all $A(x_k)$ belong to a convex hull generated by 2^n vertices (corners) because $\gamma_i \in [\gamma_i^-, \gamma_i^+]$ with $0 < \gamma_i^- < \gamma_i^+ \leq 1$. For a second order system we have $A(\gamma) = \gamma_1.A_1 + \gamma_2.A_2$ with $\gamma = [\gamma_1\ \gamma_2]^t$ where $\gamma_1 \in [\gamma_1^-, \gamma_1^+]$ and $\gamma_2 \in [\gamma_2^-, \gamma_2^+]$. The matrix $A(\gamma)$ belongs then to the convex hull $Co\{A([\gamma_1^+\ \gamma_2^+]^t), A([\gamma_1^+\ \gamma_2^-]^t), A([\gamma_1^-\ \gamma_2^+]^t), A([\gamma_1^-\ \gamma_2^-]^t)\}$. This is an important fact with respect to robust control theory (Boyd 1989).

2.2 Non-autonomous case

We will now extend the parametrization (13) to the non-autonomous case for multivariable systems. Given a linear state space model

$$\begin{aligned} x_{k+1} &= A.x_k + B.u_k \\ y_k &= C.x_k + D.u_k \end{aligned} \tag{16}$$

with the input vector $u_k \in \mathbb{R}^s$ and the output vector $y_k \in \mathbb{R}^q$. According to the Einstein summation convention this can be written as

$$\begin{aligned} x^i &:= a_j^i x^j + b_l^i u^l \\ y^i &= c_j^i x^j + d_l^i u^l \end{aligned} \tag{17}$$

In analogy with eqn (12) we can make the following parametrizations

$$\begin{aligned} x^i &:= w_{a\,j_1}^{\ i}.\frac{tanh(v_{a\,l_1}^{\ j_1} x^{l_1})}{v_{a\,l_2}^{\ j_1} x^{l_2}}.v_{a\,l_3}^{\ j_1} x^{l_3} + w_{b\,j_2}^{\ i}.\frac{tanh(t_{b\,l_4}^{\ j_2} x^{l_4})}{t_{b\,l_5}^{\ j_2} x^{l_5}}.v_{b\,l_6}^{\ j_2} u^{l_6} \\ &:= w_{a\,j_1}^{\ i}\gamma_{a\,j_1}^{\ j_1} v_{a\,l_1}^{\ j_1}.x^{l_1} + w_{b\,j_2}^{\ i}\gamma_{b\,j_2}^{\ j_2} v_{b\,l_2}^{\ j_2}.u^{l_2} \end{aligned} \tag{18}$$

and

$$\begin{aligned} y^i &= w_{c\,j_1}^{\ i}.\frac{tanh(v_{c\,l_1}^{\ j_1} x^{l_1})}{v_{c\,l_2}^{\ j_1} x^{l_2}}.v_{c\,l_3}^{\ j_1} x^{l_3} + w_{d\,j_2}^{\ i}.\frac{tanh(t_{d\,l_4}^{\ j_2} x^{l_4})}{t_{d\,l_5}^{\ j_2} x^{l_5}}.v_{d\,l_6}^{\ j_2} u^{l_6} \\ &= w_{c\,j_1}^{\ i}\gamma_{c\,j_1}^{\ j_1} v_{c\,l_1}^{\ j_1}.x^{l_1} + w_{d\,j_2}^{\ i}\gamma_{d\,j_2}^{\ j_2} v_{d\,l_2}^{\ j_2}.u^{l_2} \end{aligned} \tag{19}$$

where $\gamma_{a\,j}^{\ j}, \gamma_{b\,j}^{\ j}, \gamma_{c\,j}^{\ j}, \gamma_{d\,j}^{\ j}$ are defined like in (13). In the first equation of (18) the indices i,j_1,l_1, l_2,l_3,j_2,l_4,l_5 run from 1 to n and l_6 from 1 to s. In the first equation of (19) the indices j_1,l_1, l_2,l_3,j_2,l_4,l_5 run from 1 to n and the index i from 1 to q and l_6 from 1 to s.
We explain now how eqn. (18) and (19) can be interpreted as a neural network representation. Rewrite e.g. eqn. (18) as

$$\begin{aligned} x^i &:= w_{a\,j_1}^{\ i}.\frac{tanh(v_{a\,l_1}^{\ j_1} x^{l_1})}{v_{a\,l_2}^{\ j_1} x^{l_2}}.v_{a\,l_3}^{\ j_1} x^{l_3} + w_{b\,j_2}^{\ i}.\frac{tanh(t_{b\,l_4}^{\ j_2} x^{l_4})}{t_{b\,l_5}^{\ j_2} x^{l_5}}.v_{b\,l_6}^{\ j_2} u^{l_6} \\ &:= w_{a\,j_1}^{\ i}.tanh(v_{a\,l_1}^{\ j_1} x^{l_1}) + \frac{w_{b\,j_2}^{\ i}.v_{b\,l_6}^{\ j_2} u^{l_6}}{t_{b\,l_5}^{\ j_2} x^{l_5}}.tanh(t_{b\,l_4}^{\ j_2} x^{l_4}) \end{aligned} \tag{20}$$

where in the second term the interconnection weights of the output neurons depend on the state and the input of the system. Similar manipulations can be done on Eqn.(19). The complete neural network structure is shown in Fig.3. Observe that each matrix $A(x_k)$, $B(x_k)$, $C(x_k)$, $D(x_k)$ is expanded through a neural network.
Eqn. (18) and (19) can now be written in matrix-vector notation as

$$\begin{array}{lcccccccccc} x_{k+1} & = & W_A & \Gamma_A(x_k) & V_A & . & x_k & + & W_B & \Gamma_B(x_k) & V_B & . & u_k \\ n\times 1 & & n\times n & n\times n & n\times n & & n\times 1 & & n\times n & n\times n & n\times s & & s\times 1 \\ y_k & = & W_C & \Gamma_C(x_k) & V_C & . & x_k & + & W_D & \Gamma_D(x_k) & V_D & . & u_k \\ q\times 1 & & q\times n & n\times n & n\times n & & n\times 1 & & q\times n & n\times n & n\times s & & s\times 1 \end{array} \tag{21}$$

such that the matrices $A(x_k), B(x_k), C(x_k), D(x_k)$ in the nonlinear state space model

$$\begin{aligned} x_{k+1} &= A(x_k).x_k + B(x_k).u_k \\ y_k &= C(x_k).x_k + D(x_k).u_k \end{aligned} \tag{22}$$

can be decomposed as

$$\begin{aligned} A(x_k) &= \textstyle\sum_{i=1}^{n} \gamma_{A_i}(x_k).w_{A_i}((v_A^t)_i)^t \\ B(x_k) &= \textstyle\sum_{i=1}^{n} \gamma_{B_i}(x_k).w_{B_i}((v_B^t)_i)^t \\ C(x_k) &= \textstyle\sum_{i=1}^{n} \gamma_{C_i}(x_k).w_{C_i}((v_C^t)_i)^t \\ D(x_k) &= \textstyle\sum_{i=1}^{n} \gamma_{D_i}(x_k).w_{D_i}((v_D^t)_i)^t \end{aligned} \tag{23}$$

where $w_{.i}$ denotes the i-th column of $W_.$ and $(v_.^t)_i$ the i-th column of $V_.^t$. Remark that this is the discrete-time analog to *affine* nonlinear models in the continuous time case. A large number of systems belongs to this class, including e.g. bilinear systems (see e.g. Isidori 1985).

Distortion values $d_{A_j}(x_k), d_{B_j}(x_k), d_{C_j}(x_k), d_{D_j}(x_k)$ can be defined as

$$\begin{aligned} d_{A_j}(x_k) &= -log(\gamma_{A_j}(x_k)) \geq 0 \\ d_{B_j}(x_k) &= -log(\gamma_{B_j}(x_k)) \geq 0 \\ d_{C_j}(x_k) &= -log(\gamma_{C_j}(x_k)) \geq 0 \\ d_{D_j}(x_k) &= -log(\gamma_{D_j}(x_k)) \geq 0 \end{aligned} \tag{24}$$

If no distortion occurs one obtains a perfectly linear system (16) with matrices

$$\begin{aligned} A &= W_A V_A \\ B &= W_B V_B \\ C &= W_C V_C \\ D &= W_D V_D \end{aligned} \tag{25}$$

Observe that the number of parameters in the parametrization (18)(19) is equal to $6n^2+2ns+2nq$ (elements of W_A, V_A, W_B, V_B, T_B, W_C, V_C, W_D, V_D, T_D, where T_B and T_D contain the elements t_b and t_d in (18) and (19)). For higher order systems or in cases where the nonlinear structure of the system is known beforehand one can try to identify a *partial* parametrization like e.g.

$$\begin{aligned} x_{k+1} &= A(x_k).x_k + B.u_k \\ y_k &= C.x_k + D.u_k \end{aligned} \tag{26}$$

where the same principle can be applied for the neural representations as in Eqn. (18) and (19). Remark that like in the autonomous case again a convex hull can be specified for (18) and (19). The nonlinear models are then considered as linear systems with parametric uncertainties. This is extremely important with respect to robust controller design (see e.g. Boyd 1989, Francis 1987, Morari 1989).
In the following section we will discuss an off-line learning algorithm for the interconnection weights of the neural representation.

3 Learning algorithm for the interconnection weights

We now describe an algorithm for learning of the interconnection weights or the determination of the matrices W_A,V_A, W_B,V_B,T_B,W_C,V_C, W_D,V_D,T_D for the non-autonomous system (21). The weights are adapted in order to minimize a simulation result of interest and the initial weights are taken from linear identification results like e.g. subspace algorithms.

3.1 Cost function based on simulation error

As a criterion for learning of the interconnection weights we want to solve the following optimization problem: given a set of I/O-measurements $\{u_k\}$, $\{y_k\}$ with $k = 1, ..., N$ for a MIMO system with s inputs and q outputs

$$min\ _{W_{A,B,C,D},V_{A,B,C,D},T_{B,D}}\ J_N^{sim} = \sum_{i=1}^{q} \lambda_i . \parallel Y_i - \hat{Y}_i \parallel \tag{27}$$

taking the dynamical structure of the model into account by the constraints for $k = 1, ..., N$

$$\begin{aligned} \hat{x}_{k+1} &= W_A \Gamma_A(\hat{x}_k) V_A . \hat{x}_k + W_B \Gamma_B(\hat{x}_k) V_B . u_k \\ \hat{y}_k &= W_C \Gamma_C(\hat{x}_k) V_C . \hat{x}_k + W_D \Gamma_D(\hat{x}_k) V_D . u_k \end{aligned} \tag{28}$$

given the initial state $\hat{x}_0 = x_0$ and the number of samples N (Remark that it is also possible to take $\hat{x}_0$ as an unknown to the optimization problem (27), but this case is not considered here). The matrices Y and $\hat{Y}$ are equal to $[y_1\ y_2\ ...\ y_N]^t$ and $[\hat{y}_1\ \hat{y}_2\ ...\ \hat{y}_N]^t$ and λ is a weight vector for relative weighting of each of the output channels. The vectors Y_i and $\hat{Y}_i$ are the i-th column vector of Y and $\hat{Y}$.
In practice this optimization problem (27) can be solved by simulating the model (21) given $\{u_k\}$, x_0 and $W_{A,B,C,D}, V_{A,B,C,D}, T_{B,D}$ and to compare the simulation result $\hat{Y}$ with the output measurements Y. For local optimization methods the gradients must be calculated numerically.
In principle any norm can be chosen here for $\parallel Y_i - \hat{Y}_i \parallel$ (e.g. any p-norm). However if gradient based optimization techniques are used differentiability of the cost function is required.
In the case of the partial parametrization (26), we have the optimization problem

$$min\ _{W_A,V_A,B,C,D}\ J_N^{sim} = \sum_{i=1}^{q} \lambda_i . \parallel Y_i - \hat{Y}_i \parallel$$

with constraints for $k = 1, ..., N$

$$\begin{aligned} \hat{x}_{k+1} &= W_A \Gamma_A(\hat{x}_k) V_A . \hat{x}_k + B . u_k \\ \hat{y}_k &= C . \hat{x}_k + D . u_k \end{aligned}$$

3.2 Initial interconnection weights

The theory of estimating linear state space models from input-output measurements is already well developed. One can use e.g. subspace algorithms based on SVD or QR decompositions which have advantages with respect to structure determination, numerical robustness and computational aspects and they don't require an explicit parametrization (see e.g. De Moor 1991, Moonen 1989, Van Overschee 1992). Hence we can try to incorporate these results in the present neural network approach, by using the linear identification results in the initial neural network representation. This linearized neural network evolves then to a nonlinear model during optimization. One of the advantages of this procedure is that one can quantify the improvement of the nonlinear model to the linear model with respect to an arbitrarily chosen norm, because of starting from the linear model.
In the case of a full parametrization, we can take for the initial parameter set $W_{A,B,C,D}$, $V_{A,B,C,D}$, $T_{B,D}$

$$\begin{aligned} W_A &= \tfrac{1}{\alpha}.A && , \ V_A = \alpha . I_n \\ W_B &= I_n && , \ T_B = \alpha . I_n \ , \ V_B = B \\ W_C &= \tfrac{1}{\alpha}.C && , \ V_C = \alpha . I_n \\ W_D &= [D_{q \times s}\ O_{q \times n-s}] && , \ T_D = \alpha . I_n \ , \ V_D = \begin{bmatrix} I_s \\ O_{n-s \times s} \end{bmatrix} \end{aligned} \tag{29}$$

where α is chosen such that $E\{tanh(\alpha x_k)\} \simeq \alpha x_k$ because then the nonlinear system (16) tends to the linear system. Remark that we assume here that $n \geq s$ because of the initialization of W_D.
In the partial parametrization case one takes

$$W_A = \frac{1}{\alpha}.A \ , \ V_A = \alpha . I_n$$

4 Examples

We now illustrate the proposed algorithm on a simulated system with hysteresis and on two real life systems: a ball and beam system and a glass furnace. The optimization results in this text are obtained by the local optimization routine *fminu* of Matlab (BFGS quasi-Newton method) and simulations are performed in C (simulations in C are approximately fifty times faster than in Matlab).

4.1 Simulated system with hysteresis

As a first example we consider a simulated second order system with an hysteresis effect. Identification of the system will be studied with respect to simulation and generalization (prediction) error as a function of the complexity of the neural model (order of the model, number of hidden neurons and hidden layers) and different p norms under which the neural network model is derived.

- **Description of the system:** The system with an hysteresis effect considered here is

$$\begin{aligned} x_{1_{k+1}} &= x_{1_k} + 0.001.x_{2_k} \\ x_{2_{k+1}} &= -[300 + \beta.x_{1_k}.sign(x_{2_k})].x_{1_k} + 0.5.x_{2_k} + 0.5.u_k \\ y_k &= x_{2_k} \end{aligned} \tag{30}$$

  with input u, output y and state variables x_1, x_2 and a factor β that determines the hardness of the nonlinearity. Input-output data are generated by taking $x_{1_0} = x_{2_0} = 0$ and input data come from a white Gaussian noise process with normal distribution (zero mean and variance equal to 1) and $\beta = 200000$ in (30). The data are shown in Fig.4.

- **Linear identification:** In order to obtain a linear state space model (16) we applied the algorithm described in Moonen 1989 that consists of the following two major steps

  1. By taking SVD's of an I/O-Hankel matrix the order of the system and a state vector sequence can be estimated (see Fig.5).
  2. The A, B, C, D matrices are computed from the I/O-measurements and the state vector sequence from step 1.

  We used 400 samples (before the vertical line in Fig.5b) for identifying the model.

- **Nonlinear optimization:** For identification of a neural model from the input-output measurements we propose the partial parametrization (26), where only the A matrix is expanded through a neural network. Identification of the system will be studied with respect to simulation and generalization (prediction) error as a function of the complexity of the neural model (order of the model, number of hidden neurons and hidden layers) and different p norms under which the neural network model is derived. The simulation error $J^{sim,p}_{1\to N_1}$ is the cost function of the optimization problem (27). The notation means that points 1 till N_1 are used for identifying a model and that the cost function is measured in a certain p norm to be specified. The generalization error $J^{gen,p}_{N_2\to N_3}$ validates the derived model with respect to available data N_2 till N_3 that are outside the data set that was used for identifying the model. It measures how well the model predicts over a time horizon N_2 till N_3.

In Table 1 simulation and generalization error are compared as a function of the system order n ($n = 2, 3, 4, 7$) and different p norms ($p = 1, 2, 10, 100$) for the linear model (L) and the neural model (NL) after optimization in *iter* iterations. The model to be identified has the form

$$\begin{aligned} x_{k+1} &= W_A.tanh(V_A x_k) + B.u_k \\ y_k &= C.x_k + D.u_k \end{aligned}$$

We use as parameters $N_1 = 400$, $\alpha = 1$, $\lambda = 1$ in (27) and take initially $W_A := \frac{1}{\alpha}.A$ and $V_A = \alpha.I_n$, where A, B, C, D is the linear identification result.
In Table 2 simulation and generalization error are compared for the case of one hidden layer as a function of the number of hidden neurons. The initial interconnection matrices were chosen as

$$\begin{aligned} W_A &:= \tfrac{1}{\alpha}.[A\ O_n\ ...\ O_n] \\ V_A &:= \alpha.[I_n\ I_n\ ...\ I_n]^t \end{aligned}$$

In Table 3 the errors are compared as a function of the number of hidden layers, each consisting of n neurons, according to the model

$$\begin{aligned} x_{k+1} &= W_A.tanh(V_A^1 ... tanh(V_A^m.x_k)...) + B.u_k \\ y_k &= C.x_k + D.u_k \end{aligned}$$

with initial interconnection matrices

$$\begin{aligned} W_A &:= \tfrac{1}{\alpha}.A \\ V_A^1 &:= I_n \\ &... \\ V_A^m &:= \alpha.I_n \end{aligned}$$

Table 4 compares the distortion of two systems: system 1 with $\beta = 200000$ (high distortion) and system 2 with $\beta = 50000$ (low distortion) in (30). In both cases identification was performed for $n = 2$, $p = 2$, $N_1 = 400$ and one hidden layer. The maximum and mean values of d_1 and d_2 are given for samples 1 till 400.

- **Conclusions:** Table 1 shows that best results are obtained for the model order n as estimated from the subspace algorithms (De Moor 1991, Moonen 1989, Van Overschee 1992). For higher order models lower simulation error can be achieved but higher generalization error occurs. This is the well-known phenomenon of *overfitting*, due to non-minimality of the model.
  In Table 2 we see again the effect of *overfitting* for $3n$ and $4n$ hidden neurons, because of the increasing generalization error.
  More hidden layers, consisting each of n neurons, (see Table 3) give lower simulation error and even lower generalization error for this simulated example.
  Table 4 illustrates the capability of quantifying the nonlinear distortion on the identified model.

4.2 Identification of a ball and beam system (SISO)

We discuss here how we come from a linear state space model to a nonlinear state space model, parametrized by neural networks.

- **Description of the system:** For a description of a ball and beam system we refer to Hauser 1992. It is a SISO system with as output the position of the ball on the beam and input the angle of the beam. The system is nonlinear and of the form (26) according to Hauser 1992. Hence we expect a partial parametrization by a neural network to be sufficient for identifying the system.

- **I/O-measurements:** Input-output measurements were performed on a ball and beam system at the ESAT laboratory (see Fig.7).

- **Linear identification:** In order to obtain a linear state space model (16) we applied the same algorithm (Moonen 1989) as for the simulated system with hysteresis. We used 700 samples (before the vertical line in Fig.8b) for identifying the model. The other samples were used to validate the prediction of the model. The state space representation is

$$\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] = \left[\begin{array}{ccc|c} -0.9763 & 0.0007 & -0.0029 & -3.7230 \\ 4.3380 & 0.8673 & -0.4028 & 8.1355 \\ 7.9123 & -0.3078 & 0.0607 & 14.7046 \\ \hline -2.6521 & -1.0691 & 0.3519 & -3.1254 \end{array}\right]$$

The eigenvalues of A are $-0.9529, -0.0954, 1$.

- **Nonlinear optimization:** First we used a partial parametrization of the form (26), with $2n^2 + n(s+q) + sq = 25$ parameters. We used as parameters $N = 700$, $\alpha = 1$, $\lambda = 1$ in (27). The parameters to be optimized are the elements of W_A, V_A, B, C, D in the model

$$\begin{aligned} x_{k+1} &= W_A.tanh(V_A x_k) + B.u_k \\ y_k &= C.x_k + D.u_k \end{aligned}$$

We take initially $W_A := \frac{1}{\alpha}.A$ and $V_A = \alpha.I_3$. We optimized the simulation result both in a 2-norm and a 100-norm. In both cases the fitting ($k \leq 700$) and the prediction ($k > 700$) are improved with respect to the linear model as illustrated by Fig.8-9. The optimal neural network reprentation in the 2-norm case is

$$\left[\begin{array}{c|c} * & B \\ \hline C & D \end{array}\right] = \left[\begin{array}{ccc|c} & * & & -6.8640 \\ & & & 7.8039 \\ & & & 13.9485 \\ \hline -0.2931 & -0.3475 & -0.4903 & 4.3559 \end{array}\right]$$

where $*$ is expanded by W_A and V_A equal to

$$W_A = \left[\begin{array}{ccc} -0.0741 & -0.2612 & 0.4404 \\ 2.8532 & 1.4785 & -1.2800 \\ 6.2667 & -0.5857 & 0.7478 \end{array}\right], \; V_A = \left[\begin{array}{ccc} 0.3098 & 0.0070 & -0.0854 \\ 0.2276 & 1.5686 & 1.5931 \\ 0.5746 & 0.9498 & 1.7827 \end{array}\right]$$

In the 100-norm case we obtained as optimal result

$$\left[\begin{array}{c|c} * & B \\ \hline C & D \end{array}\right] = \left[\begin{array}{ccc|c} & * & & -3.5728 \\ & & & 7.6880 \\ & & & 14.7486 \\ \hline 0.0978 & -0.0644 & 0.0533 & 1.9486 \end{array}\right]$$

where $*$ is expanded by W_A and V_A as

$$W_A = \left[\begin{array}{ccc} -0.9550 & -0.2141 & 0.2254 \\ 4.8809 & 1.1031 & -0.9871 \\ 8.1462 & 0.2363 & -0.2541 \end{array}\right], \; V_A = \left[\begin{array}{ccc} 1.6294 & 0.0900 & 0.0752 \\ -0.3320 & 1.6375 & 0.2326 \\ -0.3411 & 0.5022 & 0.9847 \end{array}\right]$$

The cost function decreased from 1.45 in the initial linear case to 0.75 in the 2-norm case and from 0.125 to 0.079 in the 100-norm case, after 5000 iterations.
We optimized the partial parametrization further on by a full parametrization of the form (21) in the 2-norm case, where we have taken the optimal W_A and V_A from the partial parametrization and with $\alpha = 0.1$, $N = 700$ and $\lambda = 1$

$$\begin{array}{lll} W_A := W_A & , \; V_A := V_A & \\ W_B = I_n & , \; T_B = \alpha.I_n & , \; V_B = B \\ W_C = \frac{1}{\alpha}.C & , \; V_C = \alpha.I_n & \\ W_D = [D_{q\times s} \; O_{q\times n-s}] & , \; T_D = \alpha.I_n & , \; V_D = \left[\begin{array}{c} I_s \\ O_{n-s\times s} \end{array}\right] \end{array}$$

The model consists of $6n^2 + 2ns + 2nq = 66$ parameters. The cost function decreased from 0.75 to 0.71 in 5000 iterations, which is only a slight improvement as could be expected because of the physical equations described in Hauser 1992. The optimal solution is collected in the following scheme

$$\left\{\begin{array}{c|c|c||c|c|c} W_A & V_A & & W_B & T_B & V_B \\ \hline W_C & V_C & & W_D & T_D & V_D \end{array}\right\} =$$

$$\left\{\begin{array}{ccc|ccc||ccc|cccc} -0.3534 & -0.3805 & 0.6468 & 0.3363 & 0.0031 & -0.0456 & 1.2919 & -0.0004 & 0.1636 & -0.7161 & -1.2212 & -0.7372 & -7.0209 \\ 2.9565 & 1.6806 & -1.8902 & 0.0502 & 1.6800 & 1.7691 & 0.1422 & 0.9394 & -0.0072 & 0.0783 & -0.0565 & 0.2449 & 7.7142 \\ 6.2405 & -0.2238 & 0.4693 & 0.3885 & 0.8407 & 1.4739 & -0.0511 & -0.0473 & 0.9333 & -0.0439 & -0.2540 & 0.5726 & 13.9232 \\ \hline -2.9867 & -3.5163 & -4.9535 & -0.1339 & 0.1115 & 0.0458 & 4.3291 & 0.2577 & 0 & 0.0638 & 0.0429 & 0.0185 & 1.1519 \\ & & & 0.1574 & 0.2719 & -0.1565 & & & & -0.0004 & 0.1003 & 0.0658 & -0.1400 \\ & & & 0.0312 & -0.1754 & 0.2640 & & & & 0 & 0 & 0.1000 & 0 \end{array}\right\}$$

Remark that the identity matrices are relaxed to matrices with nonzero off-diagonal elements.

- **Distortion:** The 3 distortion values $d_j(x_k)$ for $j = 1, 2, 3$ are shown in Fig.11 for the case of a partial parametrization.

- **Parametric uncertainties:** According to (15) the elements of A can be plotted with respect to time and bounds on each of the elements can be derived as shown in Fig.12, respectively in the 2-norm and the 100-norm case, both in the case of partial parametrization. These bounds are extremely important for application of robust control theory (Boyd 1989, Francis 1987, Morari 1989).

4.3 Identification of a glassfurnace (MIMO)

We give a third example now on a MIMO glassfurnace system

- **Description of the system:** The system is a glass feeder of Philips Eindhoven, consisting of three inputs (two heaters and one cooler) , switched by pseudo-random binary noise and six outputs, which are temperatures in a cross-section of the oven.

- **Linear identification:** We applied again the method described in Moonen 1989. We used the first 300 samples for identification and selected a fifth order model.
- **Nonlinear optimization:** Because very good identification results are obtained by the linear identification techniques for this system (see e.g. De Moor 1991), we proposed only a partial parametrization with $2n^2 + n(s+q) + sq = 113$ parameters. The optimization is done with a 2-norm and for $N = 300$, $\alpha = 1$, $\lambda = [1\,1\,1\,1\,1\,1]^t$. The cost function decreased from 40.2 to 21.2 in 5000 iterations. Both fitting error and prediction were improved (see Fig.13).

5 Conclusions

Full and partial parametrizations for nonlinear MIMO state space models by feedforward neural networks are discussed. It is shown how concepts like distortion values and parametric uncertainties can be derived from the nonlinear models. Distortion values can be used for quantifying the "hardness" of the nonlineartity of the model. The learning of the interconnection weights is performed by optimizing a cost function that is based on simulation results and that can be expressed in any norm. We hope that the presented algorithm will open new views with respect to robust identication and control, nonlinear control and nonlinear prediction.

References

Boyd S., Q. Yang, 1989, *Structured and simultaneous Lyapunov functions for system stability problems*, Int. J. Control, Vol.49, No.6, pp. 2215-2240.

Chen S., S. Billings, P. Grant, 1990, *Nonlinear system identification using neural networks*, Int. J. Control, Vol.51, No.6, pp. 1191-1214.

Chen S., C. Cowan, S. Billings, P. Grant, 1990, *Parallel recursive prediction error algorithm for training layered neural networks*, Int. J. Control, Vol.51, No.6, pp. 1215-1228.

Cybenko G., 1989, *Approximations by superpositions of a sigmoidal function*, Mathematics of Control, Signals and Systems, 2, pp.183-192.

De Moor B., P. Van Overschee, J. Suykens, 1991, *Subspace algorithms for system identification and stochastic realization*, Recent Advances in Mathematical Theory of Systems, Control, Networks and Signal Processing, Proc. of the International Symposium MTNS-91, Kobe, Japan, MITA Press, pp. 589-595, June 17-21.

Francis B., 1987, *A course in H_∞ control theory*, New York, Springer-Verlag.

Funahashi K.-I., 1989, *On the approximate realization of continuous mappings by neural networks*, Neural networks, Vol.2, pp.183-192.

Gill P., W. Murray, M. Wright, 1981, *Practical Optimization*, Academic Press, London.

Hauser J., S. Sastry, P. Kokotovic, 1992, *Nonlinear control via approximate input-output linearization: the ball and beam example*, IEEE Transactions on Automatic Control, Vol.37, No.3, pp.392-398.

Hornik K., M. Stinchcombe, H. White, 1989, *Multilayer feedforward networks are universal approximators*, Neural Networks, Vol.2, pp.359-366.

Isidori A., 1985, *Nonlinear control systems: an introduction*, Springer, Berlin-New York.

Isidori A., A. Astolfi, 1992, *Disturbance attenuation and H_∞-control via measurement feedback in nonlinear systems*, IEEE Transactions on Automatic Control, Vol.37, No.9, pp.1283-1293.

Joshi A., 1977, *Matrices and tensors in Physics*, Wiley.

Moonen M., B. De Moor, L. Vandenberghe, J. Vandewalle, 1989, *On- and off-line identification of linear state-space models*, Int. J. Control, Vol.49, pp.219-232.

Morari M., E. Zafiriou, 1989, *Robust process control*, Prentice Hall.

Narendra K., K. Parthasarathy, 1990, *Identification and control of dynamical systems using neural networks*, IEEE Transactions on Neural Networks, Vol.1, No.1, pp.4-27.

Suykens J., B. De Moor, 1992, *Nonlinear system identification using multilayer neural networks: some ideas for initial weights, number of hidden neurons and error criteria*, ESAT-SISTA report 39, Department of Electrical Engineering, K.U. Leuven (submitted for publication).

Suykens J., J. Vandewalle, 1991, *Quasilinear approach to nonlinear systems and the design of n-double scroll ($n = 1, 2, 3, 4, ...$)*, IEE proceedings-G, Vol.138, No.5, pp. 595-603.

Van Overschee P., B. De Moor, 1992, *Two subspace algorithms for the identification of combined deterministic-stochastic systems*, ESAT-SISTA report 34, Department of Electrical Engineering, K.U. Leuven (submitted for publication).

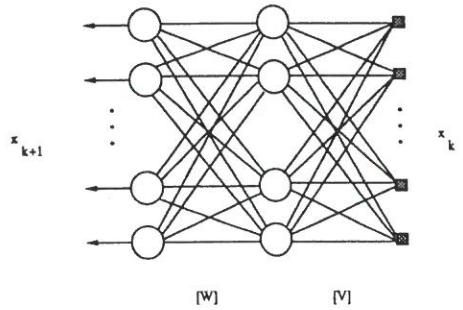


Figure 1: *Parametrization of the model $x_{k+1} = f(x_k)$ by a feedforward neural network as $x_{k+1} = W.tanh(Vx_k)$.*

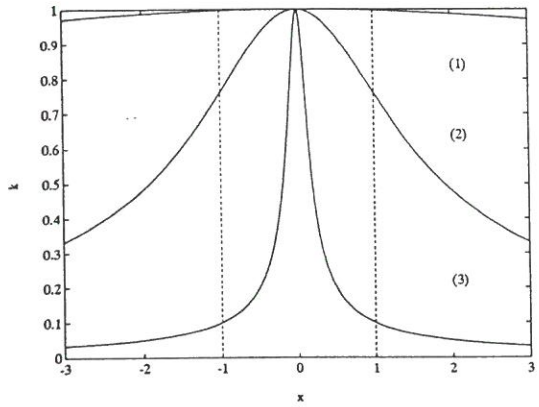


Figure 2: *Illustration of the influence of the parameter α in the function $k(x) = \frac{tanh(\alpx)}{\alpha x}$ for (1) $\alpha = 0.1$ (low distortion), (2) $\alpha = 1$, (3) $\alpha = 10$ (high distortion).*

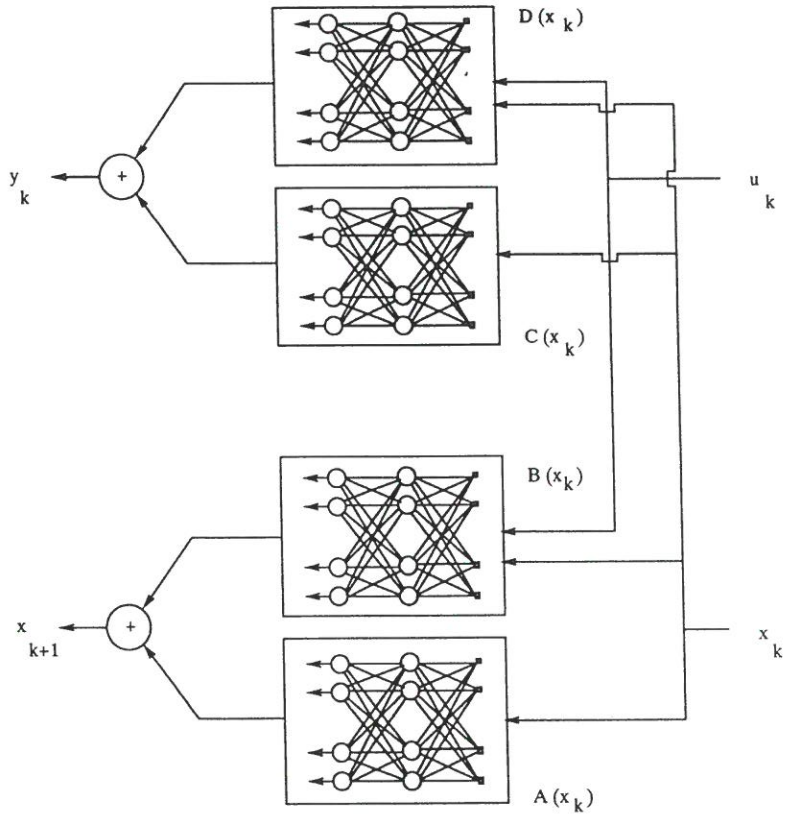


Figure 3: *Full parametrization of a nonlinear state space model by feedforward neural networks.*

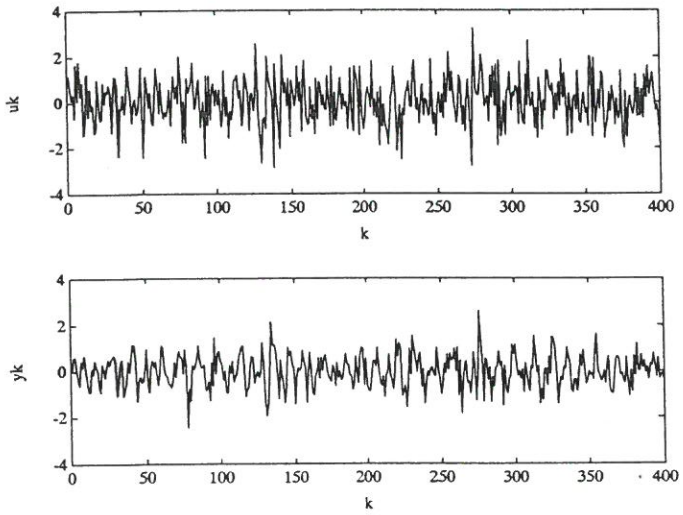


Figure 4: *Input-output data of a simulated system with an hysteresis effect.*

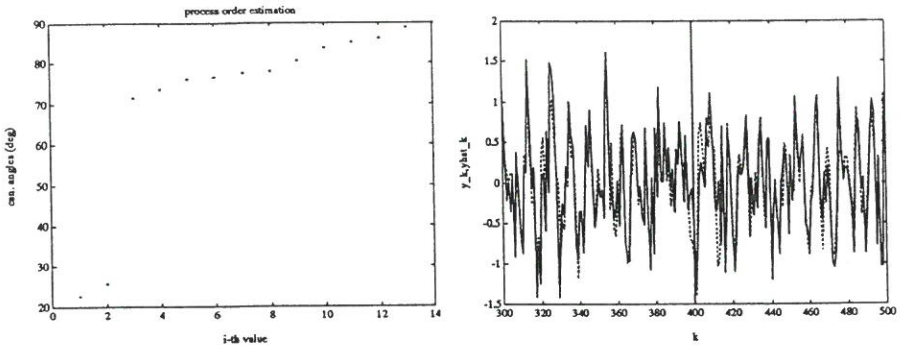


Figure 5: *Linear system identification: [Fig.a] System order estimation based on canonical angles that follow from singular value decomposition on the I/O data set, [Fig.b] Linear identification result with a second order state space model, based on two lowest canonical angles. Data before the vertical line (samples 1 till 400) were used for identification (full line = original data, dashed line = simulation result).*

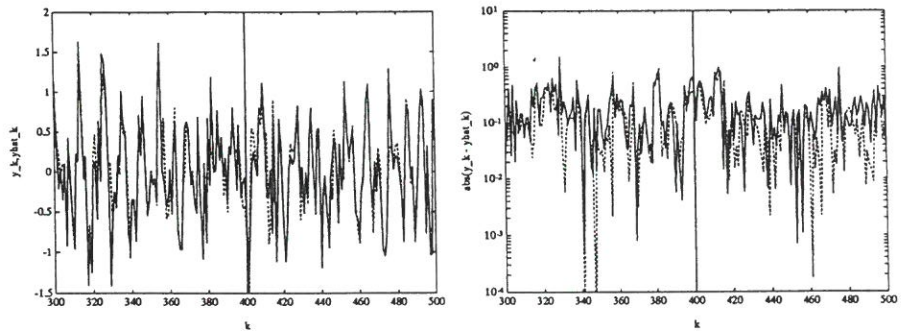


Figure 6: *Identification result after optimization of a partial neural network representation in the 1-norm case and starting with the model of Fig.5b. [Fig.a] full line = original data, dashed line = simulation result, [Fig.b] dotted line = error in the linear case of Fig.5b, dashdot line = error in the case of Fig.6a.*

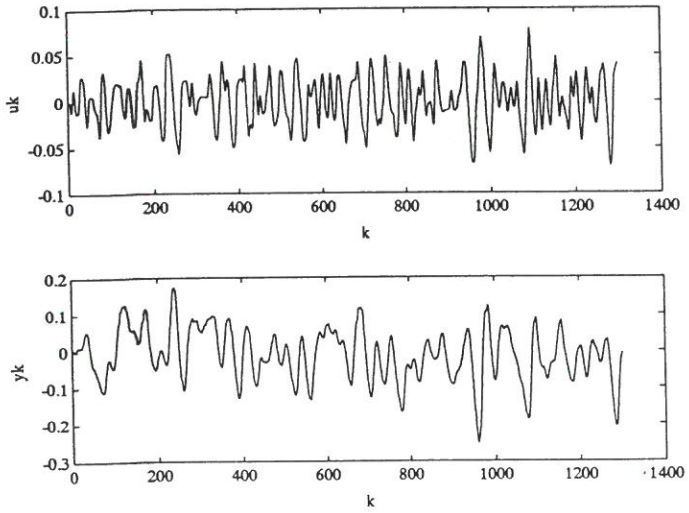


Figure 7: *Input-output measurements on a ball and beam system at the ESAT laboratory. Input u_k is the angle applied to the beam and the output y_k is the position of the ball on the beam.*

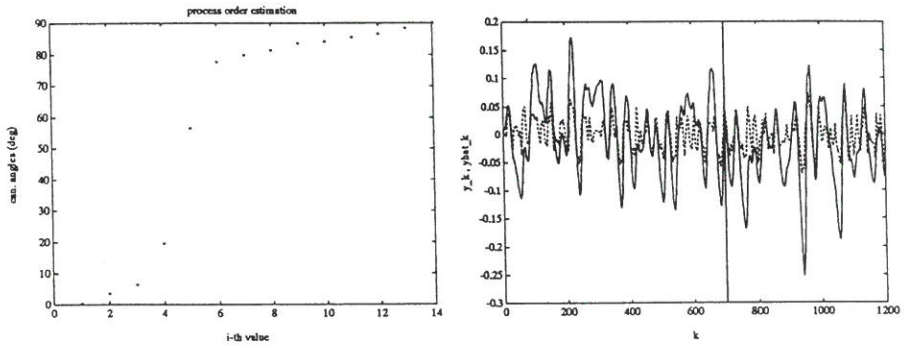


Figure 8: *Linear system identification: [Fig.a] System order estimation based on canonical angles that follow from singular value decomposition on the I/O data set, [Fig.b] Linear identification result with a third order state space model, based on three lowest canonical angles. Data before the vertical line were used for identification (full line = original data, dashed line = simulation result).*

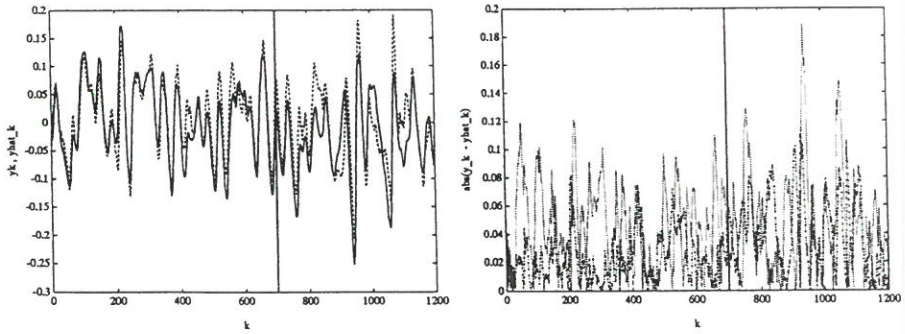


Figure 9: *Identification result after optimization of a partial neural network representation in the 2-norm case and starting with the model of Fig.8. [Fig.a] full line = original data, dashed line = simulation result, [Fig.b] dotted line = error in the linear case of Fig.8, dashdot line = error in the case of Fig.9a.*

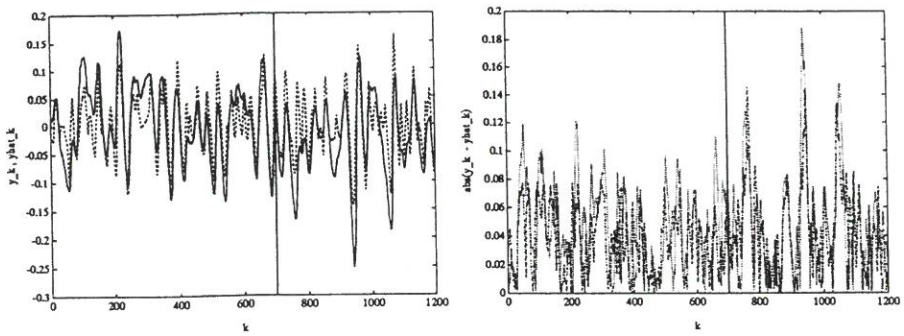


Figure 10: *Identification result after optimization of a partial neural network representation in the* 100*-norm case and starting with the model of Fig.8, [Fig.a] full line = original data, dashed line = simulation result, [Fig.b] dotted line = error in the linear case of Fig.8, dashdot line = error in the case of Fig.10a.*

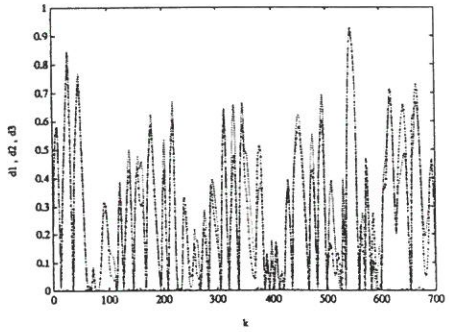


Figure 11: *Distortion values $d_j(x_k)$ through time in the* 2*-norm case of Fig.9 (dashed line = d_1, dashdot line = d_2, dotted line = d_3).*

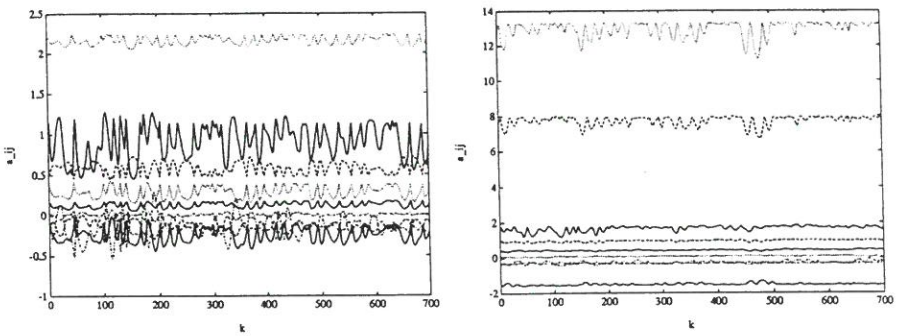
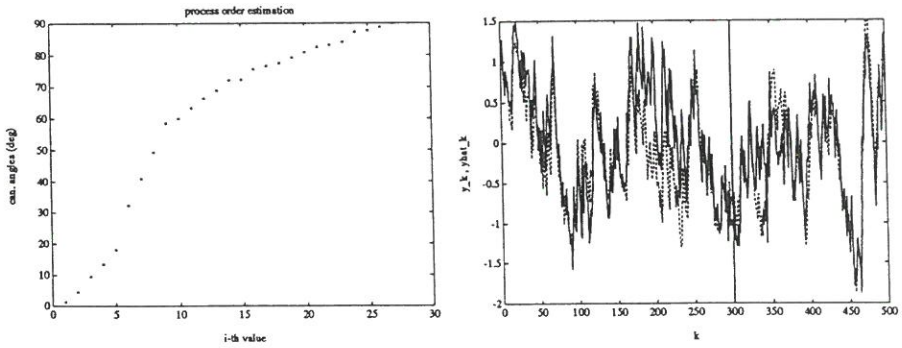


Figure 12: *Parametric uncertainties for the elements of $A(x_k)$, [Fig.a]* 2*-norm case, [Fig.b]* 100*-norm case.*



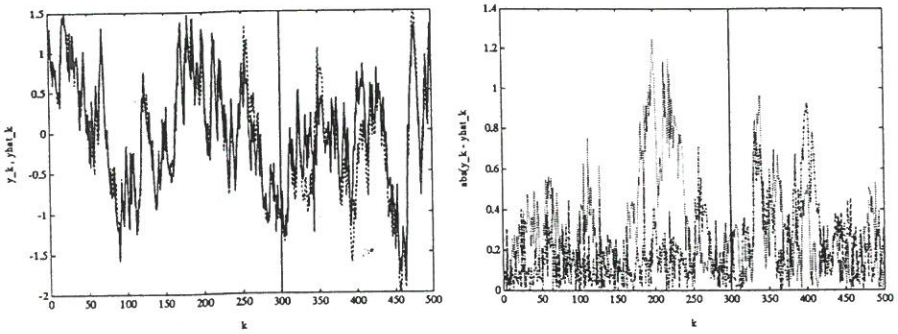


Figure 13: *Identification of a glassfurnace (MIMO system with three inputs and six outputs), [Fig.a] system order estimation based on canonical angles that follow from singular value decomposition on the I/O data set, [Fig.b] first output of linear identification result with a fifth order state space model, based on five lowest canonical angles (full line = original data, dashed line = simulation result), [Fig.c] identification result after optimization of a partial neural network representation in the 2-norm case, [Fig.d] dotted line = error in the linear case of Fig.b, dashdot line = error in the case of Fig.c.*

		n	2	3	4	7
$J^{sim,p}_{1\to400}$	p = 1	L	91.81	96.71	98.82	95.17
		NL	76.00	73.73	69.44	85.08
		iter	5303	5702	9760	7209
	p = 2	L	6.70	6.87	6.88	6.75
		NL	6.30	5.31	5.39	4.67
		iter	991	7559	10630	15678
	p = 10	L	2.16	2.08	2.03	2.02
		NL	1.54	1.53	1.18	1.18
		iter	505	2738	8451	38571
	p = 100	L	2.16	2.08	2.03	2.02
		NL	1.58	1.58	1.35	1.08
		iter	867	2289	8423	31695
$J^{gen,p}_{401\to800}$	p = 1	L	82.71	89.56	92.98	91.38
		NL	65.81	61.90	67.15	77.95
	p = 2	L	6.18	6.48	6.67	6.50
		NL	5.68	5.16	6.73	7.03
	p = 10	L	2.26	2.26	2.30	2.24
		NL	2.09	2.03	14.36	2.35
	p = 100	L	2.26	2.26	2.30	2.24
		NL	2.09	2.08	2.56	2.57

Table 1: *Simulated system with hysteresis: comparison of simulation and generalization error as a function of the system order n and four different p-norms for the linear model (L) and the neural model (NL) after optimization .*

	n_h	n	$2n$	$3n$	$4n$
$J^{sim,2}_{1\to400}$	L	6.70	6.70	6.70	6.70
	NL	6.30	5.47	4.38	4.68
	iter	991	3878	11900	14057
$J^{gen,2}_{401\to800}$	L	6.18	6.18	6.18	6.18
	NL	5.68	5.44	7.02	6.45

Table 2: *Comparison of simulation and generalization error as a function of the number of hidden neurons n_h in the one hidden layer and the case $p = 2$.*

	# h.l.	1	2	3
$J^{sim,2}_{1\to400}$	L	6.70	6.70	6.70
	NL	6.30	6.11	6.11
	iter	991	3700	7788
$J^{gen,2}_{401\to800}$	L	6.18	6.18	6.18
	NL	5.68	5.50	5.48

Table 3: *Comparison of simulation and generalization error as a function of the number of hidden layers and the case $p = 2$.*

	$[d_1^{max}\ d_2^{max}]$	$[E\{d_1\}\ E\{d_2\}]$	$[\sigma\{d_1\}\ \sigma\{d_2\}]$
system1	[0.85 0.50]	[0.22 0.12]	[0.17 0.10]
system2	[0.15 0.16]	[0.03 0.03]	[0.03 0.02]

Table 4: *Distortion values for a system with high distortion (System1) and low distortion (System2) with $E\{.\}$ the expected value over the first* 400 *samples and σ its standard deviation.*